Software Service Engineering

Software Service Engineering

WS 2019/2020 – 1. Tutorial

Valentin Siegert M.Sc.

Shovra Das M.Sc.

VSR.Informatik.TU-Chemnitz.*de*

Form:

- Detailing course knowledge
- Discussion
- Task Solving (by students, NOT by the tutor)
- Homework (voluntary)

Computer access is a must-have for this course!

News, Materials:

- http://vsr.informatik.tu-chemnitz.de/news/
- http://vsr.informatik.tu-chemnitz.de/edu/2019/sse/
- Opal: Software Service Engineering WS19/20 – Tutorial

Contact:

- valentin.siegert@informatik.tu-chemnitz.de
- shovra.das@s2018.tu-chemnitz.de
- 1/B204

1

Task 1

The programming language of the tutorial is C#.

Read any C#-Tutorial of your choice and create a simple Hello-World console application using Visual Studio.

Visual Studio Community 2019 can be downloaded from the Microsoft Azure Dev Tools Portal

of TU-Chemnitz.

http://www.tu-chemnitz.de/urz/software/dreamspark.php

- C# Characteristics
  - Very similar to Java
  - Strict type system
  - Object-oriented
  - Automatic Garbage Collection

➔Focus on robustness, durability, productivity

Data types

- Value types: int, double, bool, ...
- Reference types: string, object, Exception,...
- Generic types: List<string>, Stack<int>
- Boxing and Unboxing

```
// Value types
int i = 1;
double d = 0.25;
bool b = true;

// Reference types
string s = @"This is escaped \string";
object obj = new StringBuilder();
```

```
// Boxing and unboxing
Int32 boxedInt = i;
object boxedInt2 = i;
int unboxedInt = (int) boxedInt;
int unboxedInt2 = (int) boxedInt2;
```

Control structures

```
if(a == b)
{
   foreach (var item in items)
   {
      switch (item)
      {
         case "a":
           a++;
           break;
         default:
           b++;
           break;
      }
   }
}
```

```
for(int i=0; i<5; i++)
{
    while(i > 2)
    {
        do
        {
          Console.WriteLine("hello");
        } while (i < 3);
    }
}
```

Inheritance and interfaces

```
public interface INetwork
{
   NetworkAddress ResolveHostName(string serverName);
}

public abstract class AbstractNetwork : INetwork
{…}

public class EthernetNetwork : AbstractNetwork
{…}

public class WirelessNetwork : AbstractNetwork
{…}
```

- Class library
  - IEnumerable
  - object[]
  - List<…>
  - Dictionary<… , …>
  - Exception
  - Regex
  - Random
  - Console

The programming language of the tutorial is C#.

Read any C#-Tutorial of your choice and create a simple Hello-World console application using Visual Studio.

Visual Studio Community 2019 can be downloaded from the Microsoft Azure Dev Tools Portal

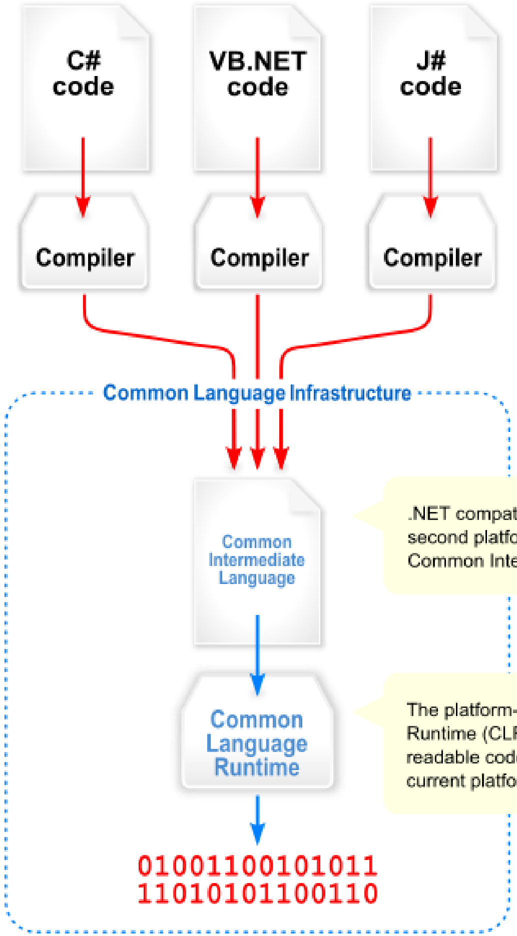
of TU-Chemnitz.

http://www.tu-chemnitz.de/urz/software/dreamspark.php

Answer the following questions:

- What are the responsibilities of the .NET Framework?
- What is common to C#, VB.NET, F#, J#?
- What are the Lambda-Expressions and LINQ?

- Development using .NET-Framework
  - Runtime Environment
    - Memory and resource management
  - Class Library
    - More than 12000 classes and datatypes
    - Grouping into namespaces
  - Tools and services
- Outcomes:
  - Console, Desktop, Web Applications
  - (Web)Services
  - Class Libraries
  - Components



.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).

The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.

Lambda-Expressions and LINQ

```
Func<int, int> transformer = x => x*x;
Func<int, int, int> transformer2 = (x,y) => x + y;
Console.WriteLine(transformer(3));
Console.WriteLine(transformer2(4,5));

IEnumerable<Point> points = GetAllPoints()
IEnumerable<string> labels = points.Where(point => point.X > 10).
                             Select(point => point.Label);


labels = from point in points
         where point.X > 10
         select point.Label;
```

2

Task 2

a) Get informed about Unit Testing and Test Driven Development (TDD). Answer the following questions:
   - What are the advantages and disadvantages of writing unit tests?
   - What is the difference between unit, integration and system tests?
   - How does the lifecycle of TDD look like?
b) Implement a simple *Calculator* application in the TDD manner. The application should enable the following computations:
   - Multiplication of two integers
   - Division of two integers

Unit Testing

- Technique to programmatically verify expected code behaviour
- Automatic check of system integrity at any time
- Facilitates clean design and separation of concerns
- But:
  - Tests are code and thus should be maintained as well
  - Testing tests is hard

Types of Tests

- Unit Tests
  - Testing of isolated code parts
- Integration Tests
  - Testing of integrated components
- System Tests
  - Verification of the system compliance with specified requirements (incl. usability, security, scalability etc.)

Test Driven Development

- A process of writing code starting from a test:
  1. Write a test that describes the behaviour of a function under the test
  2. Make sure the test **fails** (the function doesn't exist or is not implemented yet)
  3. Implement the function (do not change or edit other code)
  4. Make sure the test **passes**
  5. Perform refactoring (if needed)
  6. Make sure the test still passes

a) Get informed about Unit Testing and Test Driven Development (TDD). Answer the following questions:
   - What are the advantages and disadvantages of writing unit tests?
   - What is the difference between unit, integration and system tests?
   - How does the lifecycle of TDD look like?

b) Implement a simple *Calculator* application in the TDD manner. The application should enable the following computations:
   - Multiplication of two integers
   - Division of two integers

3

Task 3

Inform yourself about when Mock Objects should be used.

Extend the *Calculator* class from the Task 2 to write the result of the computation to a file on a local hard drive.

Use TDD and Mock Objects to simulate exceptional situations (e.g., drive is not ready or file is locked)

Mock Objects

- Simulate behaviour of real objects if they:
  - Are slow (e.g. database connections or networks)
  - Do not yet exist
  - Provide results which are not predictable or hard to reproduce (network errors)
  - Avoid placing test data into real objects

4

Homework

Create a class for computing a hash value of a string.

The hash should be computed as a rest of division of the sum of ASCII codes of all the characters in a string by 127.

Example: hash(“VSR”)=86 + 83 + 82 mod 127 = 124.

If the given string is empty, return -1.

Use TDD and Unit Testing for the development.

VSR

Your feedback on today's session:

mytuc.org/tgxs

Questions?

valentin.siegert@informatik.tu-chemnitz.de

*VSR.Informatik.*TU-Chemnitz.*de*